OPERADORES PARA COMPOSIÇÃO PARALELA (COMPOSIÇÃO SÍNCRONA E ALFABETIZADA)

Alexandre Mota & Augusto Sampaio

Operadores de Composição Paralela

- Representam a execução paralela de dois ou mais processos:
 - fluxos de controles independentes
 - interações eventuais
- Utilizados para conectar os componentes de um sistema distribuído

Operadores de Composição Paralela

- CSP oferece várias alternativas variando as condições para interação:
 - Composição paralela síncrona

Composição paralela alfabetizada

Composição paralela generalizada

Entrelaçamento

Composição Paralela Síncrona

- □ Dados os processos P e Q, O processo
 P | | Q
- executad P e Q em paralelo, sincronizando todos os eventos em comum:
 - □ a realização de um evento a de P só ocorre ao mesmo tempo que a realização de um evento a de Q
 - eventos diferentes são bloqueados (refutados)

Composição Paralela Síncrona

- - oferece os eventos iniciais comuns a ₽ e a Q,
 e espera até que haja uma comunicação
 - depois da comunicação de um evento a, comporta-se como P' | Q', onde P' e Q' comportam-se, respectivamente, como P e Q após a comunicação de a
 - □ é uma abreviação para P [|Events|] Q

Processos Paralelos e Sequenciais

```
c?x:A -> P | c?y:B -> Q
= c?z:(A \cap B) -> (P | Q)
```

- Processos paralelos podem ser expressos como processos seqüenciais
- Os operadores de composição paralela são usados por razões pragmáticas:
 - expressividade e
 - composicionalidade

Exemplo de deadlock

```
P = (a -> a -> STOP)
  [] (b -> STOP)

Q = (a -> STOP)
  [] (c -> a -> STOP)

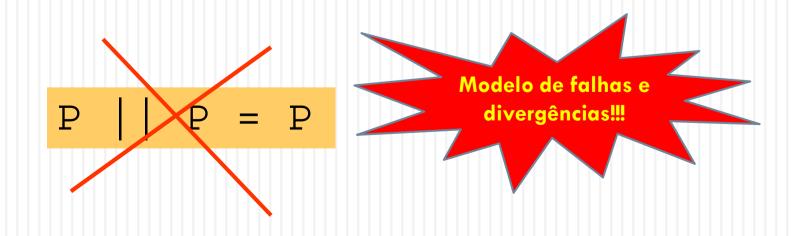
P [[Events |] Q = a -> STOP
```

```
Seja Events={a}. Então
REPEAT =
  []x: Events @ x -> x -> REPEAT
REPEATa =
  (a -> REPEAT) [|Events|] REPEAT
AS = a \rightarrow AS
           AS = REPEATa
```

```
ATM1 = incard?c -> pin.fpin(c) ->
       req?n -> dispense!n ->
       outcard.c -> ATM1
CUST1(card) =
  incard.card -> pin?p:S(card) ->
  req.50 ->
  dispense?x:\{y \mid y < - WA, y > =50\} \rightarrow
  outcard.card -> CUST1(card)
```

```
ATM1andCUST1(card)
   ATM1 [|Events|] CUST1(card)
SeqATMC(card) =
  incard.card -> pin.fpin(card) ->
  req.50 -> dispense.50 ->
  outcard.card -> SeqATMC(card)
Se fpin(x) pertence a S(x) então
 ATM1andCUST1(x) = SeqATMC(x)
```

Traces e Leis



Composição Paralela Alfabetizada

Dados os processos P e Q e os conjuntos de eventos X e Y, o processo

executa P e Q em paralelo, mas só sincronizando os eventos comuns a X e a Y

□ P (Q) só pode realizar os eventos em X (Y)

Composição Paralela Alfabetizada

- □ P [X | Y] Q
 - quando X=Y=Events é abreviado por

sendo A e B respectivamente os eventos iniciais de P e Q, oferece inicialmente o seguinte conjunto de eventos:

Processos Paralelos e Sequenciais

```
P = c?x:A \rightarrow P'
Q = c?x:B \rightarrow Q'
P [X | Y] Q
c?x:C ->
     if (x \in X) then P' else P
  [X|Y]
     if (x \in Y) then Q' else Q
```

```
CC0' = aa?x -> CC1'(x)

CC1'(x) = bb!x -> CC2'(x)

CC2'(x) = cc!x -> CC0'

[] aa?y -> CC3'(x,y)

CC3'(x,y) = cc!x -> CC1'(y)
```

Traces e Leis

```
□ traces(P [X||Y] Q) =
     \{ s \in (X \cup Y) * \mid
           s\uparrow X \in traces(P) \land
           s\uparrow Y \in traces(Q)
□ P [X | Y] (Q |~| R)
  (P[X|Y]Q) \sim (P[X|Y]R)
```

Exercícios

- Do livro texto
 - Essenciais: 2.1.2, 2.2.4
 - Opcionais: 2.1.1, 2.2.1, 2.2.2, 2.2.3